**Amendments to the Specification:**

*Please replace paragraph [1007] beginning at page 2 with the following paragraph, marked to show changes:*

[1007] One reasonable comprehensive hardware assisted profiling environment is provided by the Digital Continuous Profiling Infrastructure (DCPI) tools that run on Alpha processor systems to provide profile information at several levels of granularity, from whole images down to individual procedures and basic blocks on down to detailed information about individual instructions, including information about dynamic behavior such as cache misses, branch mispredicts and other forms of dynamic stalls. Detailed information on the DCPI tools and downloadable code may be found (at least as of the filing date) at ~~http://h30097.www3.hp.com/dcpi/~~ http://h30097.www3.hp.com/dcpi/. Additional descriptive information appears in Jennifer Anderson, Lance Berc, George Chrysos, Jeffrey Dean, Sanjay Ghemawat, Jamey Hicks, Shun-Tak Leung, Mitch Lichtenberg, Mark Vandevoorde, Carl A. Waldspurger, William E. Weihl, "Transparent, Low-Overhead Profiling on Modern Processors," in *Proceedings of the Workshop on Profile and Feedback-Directed Compilation* in conjunction with the *International Conference on Parallel Architectures and Compilation Techniques (PACT 98)*, Paris, France (October 13, 1998).

*Please replace paragraph [1036] beginning at page 8 with the following paragraph, marked to show changes:*

[1036] Figure 1 is a conceptual diagram illustrating tagging of code during code translation according to realizations of the invention. At block 107, a code translator (e.g., compiler, interpreter, etc.) begins translating a source-level code 101. The source-level code may be a function or procedure, a library file, a data object class, segment of code from an application, code for part of an application, code for an entire application, etc. The source-level code 101 may be in any source-level language including C, C++, [[Java]] JAVA, Lisp, Basic, Perl, COBOL, Fortran, etc. The code translator generates post-source-level code. The source-level code 101 may be directly translated to execution level code 105, to an intermediate level code 103 (e.g., assembly language, object code, etc.), before the execution level code 105 is generated, etc. Execution level code includes byte code, machine code, etc.

*Please replace paragraph [1095] beginning at page 27 with the following paragraph, marked to show changes:*

[1095] The described invention may be provided as a computer program product, or software, that may include a machine-readable medium having stored thereon instructions, which may be used to program a computer system (or other electronic devices) to perform a process according to the present invention. A machine readable medium includes any mechanism for storing or transmitting information in a form (e.g., software, processing application) readable by a machine (e.g., a computer). The machine-readable medium may ~~includes~~ include, but is not limited to, magnetic storage medium (e.g., floppy diskette); optical storage medium (e.g., CD-ROM); magneto-optical storage medium; read only memory (ROM); random access memory (RAM); erasable programmable memory (e.g., EPROM AND EEPROM); flash memory; electrical, optical, acoustical or other form of propagated signal (e.g., ~~carrier waves~~, infrared signals, digital signals, etc.); or other type of medium suitable for storing electronic instructions.